AMENDMENTS

In the Claims:

Please amend the claims as follows:

1. (Previously Presented) A computer system for efficiently executing instructions of computer programs, comprising:

processing circuitry having a pipeline, said pipeline configured to execute instructions from one of a plurality of programs, said processing circuitry configured to stop executing said one program during a first context switch in response to a first context switch command and to resume executing said one program during a second context switch in response to a second context switch command;

cache memory;

computer memory having a plurality of addresses; and

memory control circuitry coupled to said processing circuitry, said memory control circuitry configured to store, in response to said first context switch command, in computer memory, data written by said pipeline during execution of said one program and to store an indicator indicative of whether said data was accessed by the processing circuitry during a particular time period prior to said first context switch in executing said instructions from said one program, said memory control circuitry, in response to said second context switch command, configured to identify one of said addresses of said computer memory that is storing said indicator corresponding to said data previously written by said pipeline during execution of an instruction of said one computer program prior to said first context switch, said memory control circuitry further configured to make a determination, based on said indicator, whether said data was accessed by the processing circuitry in the particular time period prior to the first context switch for determining whether to preload said data into said cache memory in response to said second context switch, to retrieve said data, based upon

said determination, from said computer memory in response to said second context switch command, and to store said retrieved data in said cache memory based upon said indicator.

2. (Original) The system of claim 1, wherein said processing circuitry is further configured to execute instructions of another of said computer programs in response to said first context switch command.

3. (Canceled).

4. (Original) The system of claim 1, wherein said memory control circuitry is configured to maintain a plurality of mappings, each of said mappings respectively correlating data stored in said cache memory with one of said memory addresses of said computer memory, said memory control circuitry further configured to maintain a bit of information that is associated with one of said mappings, said memory control circuitry configured to assert said bit when a data value correlated with a computer memory address via said one mapping is utilized to execute an instruction of said one program, said memory control circuitry further configured to deassert said bit periodically.

5. (Original) The system of claim 4, wherein said memory control circuitry is further configured to determine, in response to said second context switch command and based on said bit, whether said data value was recently utilized by said processing circuitry to execute an instruction prior to said first context switch.

6. (Original) The system of claim 5, wherein said memory control circuitry is further configured to store said mappings and said bit to said computer memory in response to said first context switch command and to retrieve said mappings and said bit from said computer memory in response to said second context switch command.

7. (Currently Amended) A computer system for efficiently executing instructions of computer programs, comprising:

processing circuitry having a pipeline, said pipeline configured to execute instructions from one of a plurality of programs, said processing circuitry configured to stop executing said one program during a first context switch in response to a first context switch command and to resume executing said one program during a second context switch in response to a second context switch command;

cache memory;

computer memory having a plurality of addresses; and

memory control circuitry coupled to said processing circuitry, said memory control circuitry configured to maintain a plurality of mappings, said mappings respectively correlating at least one data value previously written by said pipeline during execution of an instruction and stored in said cache memory with said memory addresses of said computer memory, said memory control circuitry configured to store in said computer memory said mappings and information indicating whether said at least one data value was recently accessed and to make a determination, based on said information, whether said information indicates that said at least one data value was recently accessed prior to the first context switch, said memory control circuitry further configured to preload, into said cache memory, said at least [[on e]]<u>one</u> data value based on said determination if said information indicates that said at least one data value was recently accessed prior to said first context switch.

8. (Original) The system of claim 7, wherein said processing circuitry is further configured to execute instructions of another of said computer programs in response to said first context switch command.

9. (Canceled).

10. (Previously Presented) The system of claim 7, wherein said memory control circuitry is further configured to store said information in said computer memory in response to said first context switch command and to retrieve said information and said mappings in response to said second context switch command.

11. (Currently Amended) The system of claim [[9]]7, wherein said information has a plurality of bits respectively associated with said mappings, wherein said memory control circuitry, for each data value accessed by said memory control circuitry, is configured to assert the bit associated with the mapping that correlates said each data value with one of said computer memory addresses, and wherein said memory control circuitry is configured to periodically deassert each of said plurality of bits.

12. (Previously Presented) A method for efficiently executing instructions of computer programs, comprising the steps of:

executing a plurality of computer programs in an interleaved fashion;

switching which of said computer programs is being executed in said executing step;

storing, prior to said switching step, at an address in computer memory a data value previously written by a pipeline to a cache line in execution of an instruction corresponding

to one of said computer programs in said executing step and an indicator indicating if the cache line was recently accessed;

identifying said address in response to said switching step;

determining, based on said indicator, whether the data value was recently accessed prior to said switching step;

retrieving, based on said determining step, said data value from said address based on said identifying step and in response to said switching step if said indicator indicates that the data was recently accessed; and

storing said retrieved data value in cache memory.

13. (Original) The method of claim 12, wherein said executing step further includes the step of executing instructions of a computer program in response to said switching step, and wherein said method further comprises the steps of:

determining that said address is storing a data value previously utilized in said executing step to execute an instruction of said computer program; and

performing said identifying step based on said determining step.

14. (Original) The method of claim 12, further comprising the steps of:

correlating, respectively, data values stored in said cache memory with addresses of said computer memory;

asserting a bit each time a data value correlated with said address identified in said identifying step is accessed in response to said executing step; and

periodically deasserting said bit.

15. (Original) The method of claim 14, wherein said executing step further includes the step of executing instructions of a computer program in response to said switching step, and wherein said method further comprises the steps of:

determining, based on said bit, that said address identified in said identifying step is storing a data value previously utilized in said executing step to execute an instruction of said computer program; and

performing said identifying step based on said determining step.

16. (Currently Amended) A method for efficiently executing instructions of computer programs, comprising the steps of:

executing instructions from a computer program;

halting said executing step during a first context switch in response to a first context switch command;

resuming said executing step during a second context switch in response to a second context switch command;

maintaining a plurality of mappings;

correlating, via said mappings, data values previously written by a pipeline during the executing step and stored in a cache memory with memory addresses of computer memory outside of said cache memory;

storing said mappings in said computer memory in response to said first context switch command and information indicative of whether said data values were accessed during a particular time period prior to said first context switch;

selecting, based on said information and for preloading into the cache memory in response to the second context switch command, at least one data value from at least one of said addresses identified by said mappings; [[and]]

retrieving, based on said mappings and said selecting step, said at least one data value in response to said second context switch command if said information indicates that said at least one data value was accessed during said particular time period; and

storing said at least one retrieved data value in said cache memory.

17. (Canceled).

18. (Previously Presented) The method of claim 16, further comprising the steps of:

storing said information in said computer memory in response to said first context switch command; and

retrieving said information and said mappings in response to said second context switch command.

19. (Currently Amended) The method of claim [[17]]16, wherein said information has a plurality of bits respectively associated with said mappings, and wherein said method further comprises the steps of:

asserting each of said bits associated respectively with each of said mappings that identifies a data value accessed in response to said executing step; and

periodically deasserting each of said bits.

20. (Previously Presented) A computer system for efficiently executing instructions of computer programs, comprising:

computer memory; and

a processing unit comprising cache memory and logic configured to store in said computer memory a value indicative of whether a portion of said cache memory was recently accessed by said processor and a mapping associated with said value, said mapping indicative of a location in said computer memory storing data previously requested or previously written by an instruction of a first process being executed by the processing unit when the processing unit context switches out the first process for processing of a second process, the logic further configured to select said data for preloading in said cache memory after said second context switch if said value indicates that the data was recently accessed, retrieve said data based on said value and store said data in said cache if said value indicates that the data was recently accessed, said processing unit continuing execution of said first process with the retrieved data when the processing unit context switches out the second process and context switches in the first process.

21. (Previously Presented) A method for efficiently executing instructions of computer programs, comprising the steps of:

storing a value indicative of whether data in cache memory was recently accessed by a processing unit;

storing in said memory a mapping corresponding to said value, said mapping indicative of a location in computer memory storing said data when the processing unit context switches out the first process for processing of a second process;

determining whether said data was recently accessed prior to said processing unit switching out the first process;

preloading, based on said determining step, said data in said cache for execution of said first process by said processing unit if said value indicates that the data was recently accessed in said determining step.

22. (Previously Presented) The system of claim 21, wherein said cache memory comprises a cache line.

23. (Previously Presented) The system of claim 22, wherein said value is indicative of whether said processing unit has accessed said cache line during a particular time period.

24. (Previously Presented) The system of claim 23, wherein said value indicative of said cache memory usage is defined by a flag, said logic configured to assert said flag when said first process uses said cache line.

25. (Previously Presented) The method of claim 21, wherein said cache memory comprises a cache line.

26. (Previously Presented) The method of claim 25, wherein said value is indicative of whether said processing unit has accessed said cache line during a particular time period.

27. (Previously Presented) The method of claim 25, further comprising the step of asserting a flag when said first process uses said cache line.

28.-30. (Canceled).

31. (New) The system of claim 1, wherein said memory control circuitry stores said data and said indicator in response to said first context switch while said processing circuitry executes a second program and upon termination of execution of said second program, said memory control circuitry preloads said data, based upon said indicators, into said cache memory prior to executing said one program.

32. (New) The system of claim 1, wherein said memory control circuitry is configured to retrieve said data, based upon said determination, from said computer memory in response to said second context switch command before any instruction in said one program requests said data subsequent to said second context switch.